



بررسی عملکرد و بهبود نورون مصنوعی و پیاده‌سازی آن بر روی FPGA

بنیامین کاکاوند / گروه الکترونیک / واحد شهر قدس / دانشگاه آزاد اسلامی / تهران - ایران / b.kakavand@gmail.com
مژده مهدوی / گروه الکترونیک / واحد شهر قدس / دانشگاه آزاد اسلامی / تهران - ایران / m.mahdavi@qodsiau.ac.ir
مهدی زارع / گروه الکترونیک / واحد شهر قدس / دانشگاه آزاد اسلامی / تهران - ایران / m.zare@gmail.com

چکیده

در این مقاله با هدف شبیه‌سازی رفتار تک نورون، به پیاده‌سازی سخت‌افزاری یک نورون زیستی طبق مدل Hodgkin-Huxley پرداخته شده است که این مدل بیشترین شباهت را به نورون واقعی دارد و پیاده‌سازی آن بسیار دشوار است، ولی به کمک استفاده از قابلیت پیکربندی دوباره و اجرای دستورات به صورت موازی، طراحی سخت‌افزاری سیستم نورونی بهینه‌شده، مبتنی بر FPGA انجام شده است. برای این منظور در پیاده‌سازی تک نورون، از ایده‌های سخت‌افزاری گوناگونی، بهره گرفته‌ایم تا بهترین نوع از نورون کامل را بر اساس سرعت بالا، حجم کم، دقت بالا و قابلیت توسعه‌پذیری، طراحی کنیم. برخی از ایده‌ها در طراحی و پیاده‌سازی عبارتند از انجام محاسبات دیجیتال با بیت محدود، ساده‌سازی توابع ریاضی و تبدیل عملیات ممیز شناور به ممیز ثابت، که علی‌رغم بهینه بودن و کاهش سخت‌افزار مصرفی، از ویژگی بیشینه موازی‌سازی و دقت کافی در تولید سیگنال خروجی نیز برخوردار باشد. مدل ارائه شده در این مقاله برای بسیاری از دانشمندان در حوزه علوم اعصاب شناختی، اهمیت فراوان دارد.

کلمات کلیدی: نورون زیستی، باز پیکربندی، مدل Hodgkin-Huxley، کاهش پیچیدگی.

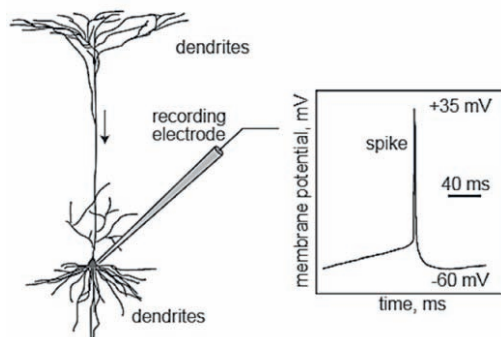
Evaluation, improvement and implementation of an artificial neuron on FPGA

Benyamin Kakavand/ Department of Electronics, Shahr-e-Qods Branch, Islamic Azad University, Tehran, Iran./ b.kakavand@gmail.com
Mojdeh Mahdavi/ Department of Electronics, Shahr-e-Qods Branch, Islamic Azad University, Tehran, Iran/ m.mahdavi@qodsiau.ac.ir
Mahdi Zare/ Department of Electronics, Shahr-e-Qods Branch, Islamic Azad University, Tehran, Iran/ m.zare@gmail.com

ABSTRACT

In this paper, simulation the behavior of a single neuron, and the implementation of the hardware of a biochemical neuron according to the Hodgkin-Huxley model is investigated, that is closely resembles the real neuron, and its implementation is very difficult, But with the help of re-configuration and implementation on parallel, the FPGA-based hardware design of the optimized neuronal system was performed. For this purpose, we have been using a variety of hardware ideas for the implementation of single-neurons to design the best type of complete neuron based on high speed, compactness, high precision and expandability. Some of the ideas in design and implementation include doing limited digital calculations, simplifying mathematical functions, and converting floating-point operations to fixed-point, which have the optimization and reduction of consumable hardware, have the maximum parallelism and the precision of the production of the output signals. The model presented in this paper is very important for many scientists in the field of cognitive neuroscience.

Keywords: Bio-neuron, reconfiguration, Hodgkin-Huxley model, reduced complexity.



شکل ۱: تغییرات سطح ولتاژ نورون در یک اسپایک بر حسب زمان [۱۰]

مدل نورونی H-H کامل‌ترین مدل نورونی زیستی است و رفتارهای یک نورون واقعی را از خود نشان می‌دهد. بنابراین می‌توان از آن به‌عنوان مدل مناسب برای شبیه‌سازی رفتارهای مغز استفاده کرد. از آن جمله می‌توان به مقاله [۱۵] اشاره کرد که در آن نویسنده، شبکه نورونی از نورون‌های مدل Hodgkin-Huxley (H-H) را با استفاده از ASIC پیاده‌سازی کرده است، اگرچه پیاده‌سازی خاص منظوره می‌تواند مزایای خود را داشته باشد، ولی در این حالت بعد از ساخت، امکان تغییر سیستم وجود ندارد. در [۱۶]، گرس و همکارانش نورون مدل H-H را بر روی FPGA پیاده‌سازی کرده‌اند. برای پیاده‌سازی این مدل، ابتدا مدل را در محیط Simulink شبیه‌سازی کرده و توسط مولد سیستم مدل را به کد قابل انتقال بر روی FPGA تبدیل کرده‌اند. اگرچه در این روش، پیاده‌سازی راحت‌تر و دارای زمان کمی است، اما واضح است که سیستمی که با روش مبتنی بر HDL طراحی شود، انعطاف‌پذیری بیشتری در بهبود طراحی و پیاده‌سازی بر روی انواع تراشه‌های FPGA خواهد داشت. پوره‌هاج و تنگ در [۱۷] به پیاده‌سازی نورون مدل H-H پرداخته‌اند. آن‌ها برای پیاده‌سازی این مدل، از همان مرحله نخست به تبدیل و گسسته‌سازی معادلات مربوطه با استفاده از روش اویلر پرداختند. اگرچه این نوع پیاده‌سازی مصرف منابع سخت‌افزاری پایینی دارد و دارای سرعت اجرایی بالا است، ولی دقت کمی دارد در حالی که در کاربردهای پزشکی دقت مهمترین معیار است. در [۱۸-۲۰] یک شبکه نورونی زیستی بر مبنای نورون‌های مدل H-H پیاده‌سازی شده است. در این پیاده‌سازی‌ها از چندین مدار مجتمع آنالوگ که محاسبات مربوط به هر نورون در این مدارهای مجتمع انجام می‌شود، استفاده شده است. در این سیستم‌ها، به دلیل پیاده‌سازی نورون با مدارهای مجتمع، امکان تغییر مدل و یا تغییر اتصالات بعد از ساخت وجود ندارد. گریسیا و همکارانش در [۲۱] شبکه نورونی پیاده‌سازی شده در [۲۰] را برای بررسی رفتارهای مختلف نورون‌ها در شبکه نورونی استفاده کرده‌اند. آن‌ها به‌منظور کم کردن سخت‌افزار مصرفی، ثابت زمانی موجود در معادلات H-H را به‌صورت یک مقدار ثابت فرض کردند. اگرچه این کار باعث کم شدن فضای مصرفی می‌شود، ولی در مصارفی که نیاز به دانستن تغییرات این ثابت زمانی باشد، پاسخ‌گو نیست. مدل Izhikevich را می‌توان بین مدل‌های نورون زیستی و انتزاعی با سخت‌افزار بالا و دقت متوسط قرار داد، که در مقالات [۲۴-۲۲] به پیاده‌سازی این مدل پرداخته شده است. از بین تمامی مدل‌های ارائه شده برای نورون، مدل H-H به‌عنوان یک مدل مناسب و کامل برای شبیه‌سازی رفتار نورون انتخاب شده است [۲۱-۱۶]. تاکنون تمایل کمتری به پیاده‌سازی مدل H-H به دلیل پیچیدگی پیاده‌سازی و فضای مصرفی

نورون‌ها سلول‌های تحرکی و مهارتی هستند که سیگنال‌ها را از نورون‌های دیگر دریافت نموده، داده‌ها را پردازش کرده و آنها را به کمک یک فرآیند پیچیده الکتروشیمیایی منتقل می‌کنند. سازماندهی مغز و فعل و انفعالات بین نورون‌ها بر اساس تابعیت نورون‌ها و محل آناتومیک آن‌ها تغییر می‌کند. در یک شبکه نورونی هزاران نورون به‌صورت موازی با یکدیگر در ارتباط هستند [۱]. در دهه کنونی، شبیه‌سازی رفتار نورون، در یک شبکه نورونی بزرگ به یک علاقه خاص تبدیل شده است؛ به‌طوری که در سال‌های اخیر از شبکه‌های نورونی در زمینه‌های مختلف پزشکی مانند بررسی اثر بیماری اتیسم و یا تاثیر آن بر سیستم درک انسان استفاده شده است [۴-۱]. تعامل پویا بین نورون‌ها در عملکرد شبکه نورونی بسیار مهم است. چنین تعاملی در سیستم با ماهیت سریال، قابل شبیه‌سازی و پیاده‌سازی نیست، بنابراین مدل‌سازی موازی، برای تعاملات پویای نورون‌ها ضروری بوده و پیاده‌سازی یک سیستم نورونی زیستی در یک بستر موازی به‌منظور دستیابی به یک شبکه با تعداد نورون‌های زیاد و زمان اجرای کم قابل اجرا است [۵]. ویژگی اجرای موازی نورون‌ها در یک شبکه نورونی بر روی نرم‌افزار ناممکن است [۶]. از بین روش‌های پیاده‌سازی سخت‌افزار، پیاده‌سازی دیجیتال ترجیح داده می‌شود زیرا ویژگی‌هایی چون سیگنال به نویز کمتر، تکرارپذیری راحت‌تر، انعطاف‌پذیری زیاد و قابلیت تست راحت‌تر نسبت به پیاده‌سازی آنالوگ را داراست [۵]. به‌علاوه، زمان تولید نمونه اولیه و هزینه مهندسی طراحی دیجیتال کمتر است [۷]. پیاده‌سازی به کمک پردازنده‌های خاص منظوره به‌دلیل ویژگی اجرای ترتیبی آن روش مناسبی برای مدل کردن رفتار موازی نورون‌ها در یک شبکه نورونی نیست. پیاده‌سازی‌های مبتنی بر ASIC علیرغم سرعت مناسب اجرا، توان مصرفی پایین و هزینه تولید نهایی اندک، قابلیت پیکربندی دوباره را ندارد [۸]. طرح‌های مبتنی بر FPGA دارای مزایای قابلیت پیکربندی دوباره و پردازش موازی است و بدین جهت، سخت‌افزار بهینه برای پیاده‌سازی شبکه نورونی به‌شمار می‌رود [۵]. از جمله مزایای پیاده‌سازی بر روی FPGA این مزیت است که کد طراحی شده با HDL، مختص یک نوع FPGA خاص نیست و به راحتی قابل انتقال از یک نوع تراشه به نوع دیگر است [۹].

هر نورون از یک هسته و بدنه نورون تشکیل شده است. نورون‌ها اطلاعات را از طریق شاخه‌هایی به‌نام دندریت دریافت کرده و آن‌ها را به بدنه منتقل می‌کنند. این اطلاعات از آکسون به پایانه آکسون رسیده و به نورون‌های دیگر منتقل می‌شود. به این ترتیب، نورون‌ها اطلاعات را تا فاصله‌های دور منتقل می‌کنند [۱۰]. فعالیت الکتریکی نورون‌ها از طریق نقل و انتقال یون‌ها صورت می‌گیرد [۱۰]. اگر نورون توسط یک جریان خارجی و یا توسط نورون‌های دیگر تحریک شود، تعادل یون‌ها به هم ریخته و جریان‌های یونی از داخل به خارج غشاء و برعکس برقرار می‌شود. در اثر این تحریک، پتانسیل نورون به یک مقدار مثبت افزایش پیدا می‌کند و بعد از مدت کوتاهی به مقدار حالت استراحت خود بر می‌گردد. در این حالت گویند که نورون آتش کرده است و یا اسپایک زده است، شکل (۱) نمودار آتش کردن نورون را در اثر تحریک نشان می‌دهد. در طراحی نورون هر چه به سمت مدل‌های انتزاعی پیش برویم، بهره‌وری محاسباتی افزایش می‌یابد، نمونه‌هایی از مدل انتزاعی در مراجع [۱۴-۱۱] مورد استفاده قرار گرفته است. اگرچه این نوع شبکه‌ها در کاربردهای خاص، چون تشخیص الگو و... بسیار مورد استفاده قرار می‌گیرد، ولی نمی‌توان از آن انتظار رفتار یک نورون زیستی را داشت.

داده شده است. دریچه مربوط به کانال این یون از چهار پره تشکیل شده است که با توان چهار در متغیر n نشان داده می‌شود.

$$I_K = \bar{g}_K n^4 (V - V_K) \quad (3)$$

در مدل H-H جریان‌های ناشی از سایر یون‌ها، از جمله یون کلسیم به صورت یک جریان نشتی (IL) نشان داده می‌شود. این جریان در رابطه (۴) نشان داده شده است.

$$I_L = g_L (V - V_L) \quad (4)$$

مقادیر پارامترهای مربوط به دریچه‌های کانال‌های یونی با استفاده از معادلات (۵) تا (۱۳) بدست می‌آید [۲۷]. روابط (۵) تا (۷) نشان‌دهنده معادلات مربوط به پارامتر m است.

$$\alpha_m = \frac{0.182(V - 35)}{V + 35} \frac{1}{1 - e^{-\frac{V + 35}{9}}} \quad (5)$$

$$\alpha_h = 0.25e^{-\frac{V + 90}{12}} \quad (6)$$

$$\alpha_N = \frac{0.02(V - 25)}{1 - e^{-\frac{V - 25}{9}}} \quad (7)$$

روابط (۸) تا (۱۰) نشان‌دهنده معادله مربوط به دریچه غیرفعال‌ساز کانال یون سدیم است. با صرف نظر کردن از دینامیک h، روابط H-H از مرتبه چهار به مرتبه دو کاهش یافته است.

$$\beta_h = \frac{0.25e^{-\frac{V + 62}{12}}}{e^{-\frac{V + 90}{12}}} \quad (8)$$

$$\beta_m = \frac{0.125(-V + 35)}{1 - e^{-\frac{V + 35}{9}}} \quad (9)$$

$$\beta_N = \frac{0.002(-V - 25)}{1 - e^{-\frac{V - 25}{9}}} \quad (10)$$

روابط مربوط به پارامتر کانال پتانسیمی در روابط (۱۱) تا (۱۳) نشان داده شده است.

$$\frac{dn}{dt} = \alpha_n(1 - n) - \beta_n n \quad (11)$$

$$\frac{dh}{dt} = \alpha_h(1 - h) - \beta_h h \quad (12)$$

$$\frac{dm}{dt} = \alpha_m(1 - m) - \beta_m m \quad (13)$$

پارامترهای C، V، T، I، g و به ترتیب نشان‌دهنده چگالی خازنی، ولتاژ، زمان، جریان تحریکی و چگالی رسانایی است. واحدهای استفاده شده برای این پارامترها برابر است با:

$$\mu\text{A}/\text{cm}^2, \text{ms}/\text{sm}^2, \text{ms}, \mu\text{mV}, \mu\text{F}/\text{Cm}^2$$

مقادیر پارامترهای مورد استفاده در معادلات بیان شده، برابر است با $C=1$ ، $g_{Na}=100$ ، $V_{Na}=50$ ، $g_K=80$ ، $V_K=-100$ ، $g_L=1.0$ و $V_L=-67$. این مقادیر همانند مقادیر استفاده شده در [۲۸، ۲۷] است.

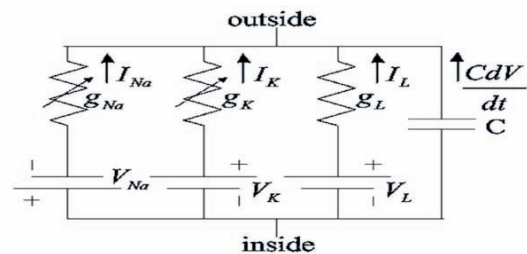
۲-۱ معماری پیشنهادی برای پیاده‌سازی تک نورون براساس مدل H-H

برای پیاده‌سازی تک نورون معماری شکل (۳) مورد استفاده قرار گرفته است. هر کدام از بلوک‌های موجود در این معماری، نشان‌دهنده

زیاد آن وجود داشته است [۲۵]. قابلیت پیکربندی دوباره و اجرای موازی برای پیاده‌سازی شبکه‌های نورونی در انتخاب سخت‌افزار مناسب بسیار حائز اهمیت است. در بین بسترهای موجود برای طراحی دیجیتال، FPGA هر دو ویژگی را دارد. بنابراین به‌عنوان بستر مناسب برای این پژوهش انتخاب شده است. از مدل تک نورون بهبود یافته در این مقاله می‌توان در راستای ساخت یک بستر سخت‌افزاری مناسب برای یک نورون با کانال‌های ورودی و خروجی فراوان استفاده کرد که با الهام از ساختار مغز بتواند قابلیت‌های مغز نظیر حجم کوچک، سرعت پردازش زیاد، مصرف توان کم و قابلیت تشخیص و استنتاج چشمگیری را در خود داشته باشد. در ادامه و در بخش دوم مدل نورون مذکور بیان می‌شود. در بخش سوم الگوریتم پیشنهادی برای بهینه‌سازی هر کدام از اجزای موردنیاز توضیح داده می‌شود، در بخش چهارم نتایج شبیه‌سازی، در بخش پنجم گزارش پیاده‌سازی و در انتها نیز نتیجه مقاله آمده است.

۲-۲ مدل نورون H-H

همان‌گونه که در بخش قبل اشاره شد، مدل H-H برای نورون توسط Hodgkin-Huxley پس از آزمایشات بسیار آرایه گردید که با این توصیف، توانستند جایزه نوبل را از آن خود کنند. مهمترین ویژگی این مدل، الهام گرفتن از ساختار بیولوژیکی نورون واقعی است. غشا نورون مانند یک سد، یون‌های داخل نورون را از مایع خارجی جدا می‌کند. این عمل می‌تواند توسط یک خازن در مدارهای الکتریکی جایگزین شود. بنابراین یک خازن با ظرفیت C می‌تواند نقش غشا سلول را برای مدل‌سازی بازی کند. مدل نورونی استفاده شده در این پیاده‌سازی، مدل مرتبه چهار است. دو نوع نورون تحریکی و مهاری در شبکه وجود دارد که ساختار مدل نورون برای هر دو نوع نورون تحریکی و مهاری یکسان است. مدار شکل (۲) برای مدل‌سازی روابط نورون توسط هاجکین و هاگکلی معرفی شده است [۲۶].



شکل ۲: مدار پیشنهادی توسط هاجکین و هاگکلی در یک اسپایک [۲۶]

با توجه به [۱۰] و [۲۶، ۲۷] رابطه بین پتانسیل غشاء و جریان‌های یونی و جریان تحریکی در معادله (۱) نشان داده شده است.

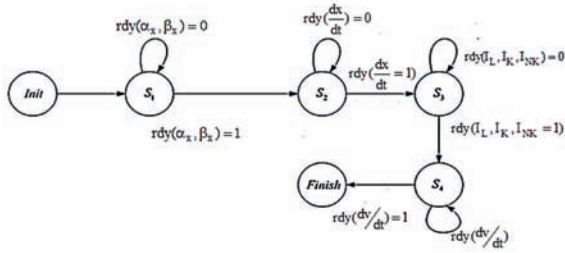
$$\frac{CdV}{dt} = I - I_{Na} - I_K - I_L \quad (1)$$

در عبارت (۱) I_{Na} نشان‌دهنده جریان یونی مربوط به سدیم است. این جریان تحت تاثیر دو متغیر دریچه فعال‌ساز (m) و غیرفعال‌ساز (h) است. فعال‌ساز این دریچه از سه پره تشکیل شده است که با توان سه در m اثر می‌گذارد و سرعت باز شدن این دریچه بیشتر از سرعت بسته شدن دریچه توسط h است. رابطه (۲) نشان‌دهنده این جریان یونی است.

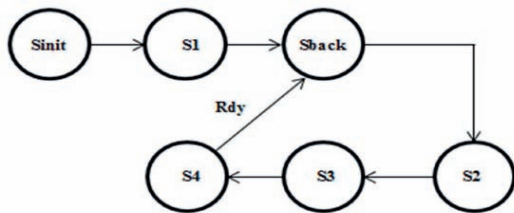
$$I_{Na} = \bar{g}_{Na} m^3 h (V - V_{Na}) \quad (2)$$

جریان پتانسیمی ناشی از حرکت یون‌های پتانسیم در رابطه (۳) نشان

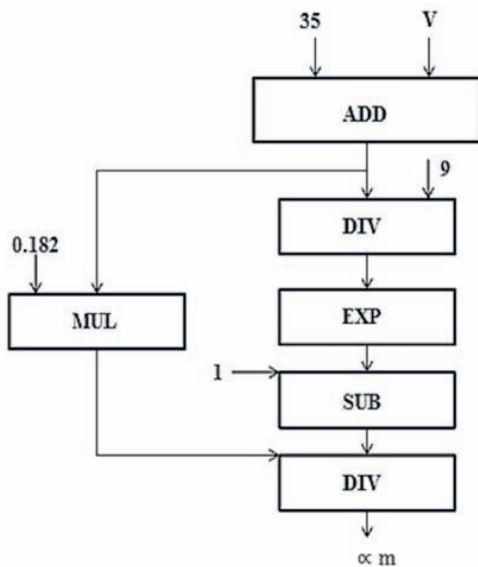
این زمینه ایده‌های مختلفی در پیاده‌سازی توابع در این مقاله به کار گرفته شده است، که در ادامه توضیحات مربوط به آن‌ها شرح داده خواهد شد.



شکل ۴: کنترلر پیشنهادی برای کنترل عملیات بین بلوکها



شکل ۵: کنترلر طراحی شده برای تک نورون



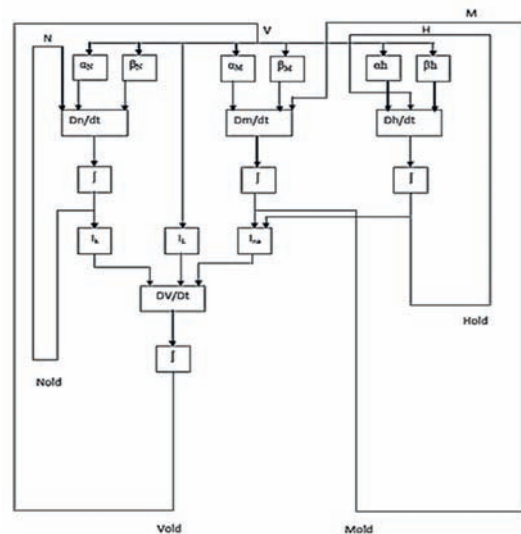
شکل ۶: بلوک دیاگرام مدار پیاده‌سازی شده برای a_m

۳- الگوریتم پیشنهادی

در پیاده‌سازی سخت‌افزاری باید نحوه نمایش داده‌ها و تعداد بیت‌های لازم برای نمایش انتخاب شود. برای تصمیم‌گیری در مورد تعداد بیت مناسب برای پیاده‌سازی بهینه با دقت بالا، مراحل زیر به ترتیب انجام شد:

- شبیه‌سازی تک نورون توسط Simulink به ازای بازه پیوسته‌ای از ورودی‌ها و بدست آوردن بازه تغییر تک‌تک متغیرها، و در نهایت تعیین تعداد بیت مورد نیاز برای قسمت اعشار با استفاده از بازه

معادلات مربوط به تک نورون هستند که در روابط (۱) تا (۱۳) آورده شده است. کنترل ارتباط بین این بلوک‌ها، زمان انتقال ورودی به هر کدام، زمان آماده شدن خروجی و تمامی فعالیت‌های مربوط به سیستم کنترل به‌طوری‌که از درستی انجام عملیات اطمینان حاصل شود، توسط کنترلر انجام می‌شود، که ماشین حالت آن در شکل (۴) نشان داده شده است. در ابتدا تمام بلوک‌ها ریست می‌شوند و کنترلر به مرحله بعدی، S1 تغییر حالت می‌دهد. در حالت S1 کلاک مربوط به بلوک‌های α_x و β_x که شامل پارامترهای m, n, h می‌شود فعال شده و این مدارها کار خود را شروع می‌کنند و تا زمانی که خروجی آنها آماده شود، کنترلر در این حالت باقی می‌ماند. بعد از آماده شدن خروجی این بلوک‌ها، کنترلر با آمدن کلاک به حالت S2 تغییر حالت می‌دهد. در حالت S2 کلاک بلوک‌های dx/dt که شامل m, n, h می‌شود، فعال شده و کنترلر تا زمانی که خروجی آماده نشده، در این حالت باقی می‌ماند. در S3 کلاک جریان، I_L و I_{Na} فعال شده و با آماده شدن خروجی آن‌ها، کنترلر به حالت S4 یعنی S4 تغییر حالت می‌دهد. در S4، کلاک dv/dt فعال شده و با آماده شدن خروجی، کنترلر به حالت Finish رفته و پاسخ یک مرتبه اجرای نورون آماده می‌شود. کنترلر شکل (۵) که ارتباط بین بلوک نورونی و انتگرال‌گیرها را ممکن می‌سازد، از حالت Sinit کار خود را شروع می‌کند و با آمدن کلاک به حالت S1 تغییر حالت می‌دهد. در حالت S1 کلاک انتگرال‌گیر فعال می‌شود که با مقادیر اولیه مقدردهی شود. در حالت Sback که حالت بازگشت برای تکرار عملیات نیز است، کلاک انتگرال‌گیر غیرفعال شده، سپس کنترلر به حالت S3 می‌رود و تا زمانی که خروجی بلوک نورونی آماده شود، در این حالت باقی می‌ماند.

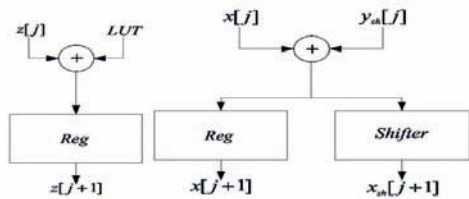


شکل ۷: معماری پیشنهادی برای پیاده‌سازی تک نورون

با آماده شدن خروجی بلوک نورونی، کنترلر به حالت S4 منتقل شده و کلاک انتگرال‌گیر فعال می‌شود که داده‌های جدید را دریافت کند. شکل (۶) بلوک دیاگرام مدار پیاده‌سازی شده برای معادله α_m را نشان می‌دهد. با توجه به بلوک دیاگرام شکل (۳) و شکل (۶)، برای پیاده‌سازی تک نورون و معادلات آن باید در ابتدا تمام توابع مورد نیاز، جمع/تفریق، ضرب/تقسیم، تابع نمایی، توان و انتگرال‌گیر، با تعداد بیت مناسب پیاده‌سازی شوند، به‌طوری‌که نتیجه نهایی از دقت مناسب برخوردار باشد و از طرفی مساحت زیادی را اشغال نکند. در

تغییر متغیرها.

استفاده شد، جدا کردن بخش صحیح از بخش اعشار ورودی تابع نمایی، محاسبه جداگانه هر بخش و در نهایت حاصلضرب دو پاسخ بدست آمده برای محاسبه پاسخ تابع نمایی است.

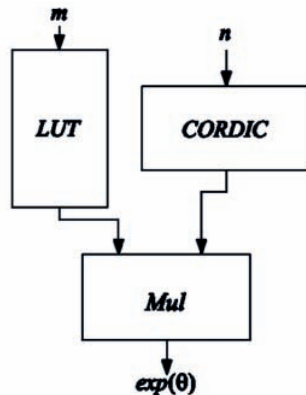


شکل ۷: بلوک دیاگرام پیشنهادی برای پیاده‌سازی الگوریتم CORDIC

$$\exp(\theta) = \exp(m) \cdot \exp(n) \quad (21)$$

$$\theta = m + n \quad (22)$$

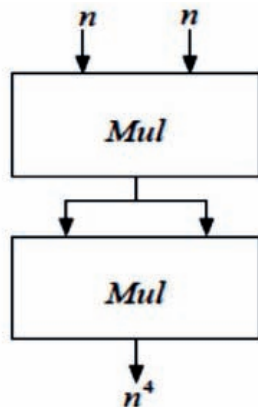
معماری پیشنهادی در شکل (۸) نشان داده شده است. با استفاده از این پیاده‌سازی، پاسخ دقیق تابع نمایی، در زمان کم و با مصرف کمینه سخت‌افزار بدست آمد.



شکل ۸: بلوک دیاگرام پیشنهادی برای پیاده‌سازی تابع نمایی

۳-۲- پیاده‌سازی دیجیتال تابع توان

با توجه به روابط (۲) و (۳) برای پیاده‌سازی جریان‌ها نیاز به محاسبه توان سه و توان چهار است. برای پیاده‌سازی توان، معماری شکل (۹) در نظر گرفته شده است. به این صورت تعداد ضرب‌کننده‌ها کاهش یافته و به سرعت بالا و دقت مناسب دست یافتیم.



شکل ۹: بلوک دیاگرام محاسبه‌گر توان

محدود کردن تعداد ارقام اعشار تمامی پارامترها و عملیات موجود در کد تک نرون شبیه‌سازی شده در Simulink، تغییر تعداد ارقام اعشار و بررسی تغییرات حاصل در نتیجه نهایی و فرکانس آتش کردن نرون‌ها و بدست آوردن تعداد بیت‌های مورد نیاز برای نمایش ارقام در پیاده‌سازی دیجیتال.

از جمله توابعی که در پیاده‌سازی آن‌ها از روش‌های محاسبات سخت‌افزاری برای بهبود پیاده‌سازی استفاده شده است می‌توان به تابع نمایی اشاره کرد که محدوده ورودی آن با صرف حداقل سخت‌افزار افزایش می‌یابد. همچنین تابع توان که در پیاده‌سازی آن علاوه بر داشتن دقت کافی، تعداد ماژول‌های مورد استفاده کاهش یافت. معیارهای انتخاب روش پیاده‌سازی توابع عبارتند از: استفاده از توابع موردنظر در چند بخش مدار، تعداد ارقام مورد استفاده در عملیات، تعداد کلاک پالس‌هایی که می‌توان برای تولید خروجی بعد از اعمال ورودی منتظر ماند، اولویت سرعت یا حجم سخت‌افزاری و بازه تغییرات ورودی. در ادامه به بررسی پیاده‌سازی توابع و ایده‌های ارایه شده می‌پردازیم.

۳-۱- پیاده‌سازی دیجیتال تابع نمایی

با توجه به روابط (۵) و (۶)، تابع نمایی از جمله توابع مهم و پرکاربرد در پیاده‌سازی تک نرون است و دقت خروجی آن تاثیر زیادی در دقت خروجی نهایی دارد. برای پیاده‌سازی این تابع روش‌های مختلفی چون پیاده‌سازی با روش اویلر، پیاده‌سازی بر اساس جدول و پیاده‌سازی با استفاده از الگوریتم CORDIC وجود دارد. روش اویلر به دلیل مصرف سخت‌افزار زیاد و دقت کم در خروجی روش مناسبی نیست. ایده به کار رفته در این مقاله، تلفیقی از دو روش پیاده‌سازی بر اساس جدول و الگوریتم CORDIC به منظور دست یافتن به پیاده‌سازی بهینه، سرعت اجرای زیاد و دقت بالا است. الگوریتم CORDIC به دلیل بهینه بودن آن در پیاده‌سازی توابعی چون: هائپربولیک، Sin و ... کاربرد دارد. اساس کار این الگوریتم با استفاده از جمع و شیفتهای پیاپی است که تا رسیدن به جواب مطلوب انجام می‌شود. بدین ترتیب، با حداقل سخت‌افزار، خروجی با دقت کافی تولید می‌شود [۲۹]. روابط (۱۴) تا (۲۰) نشان‌دهنده این الگوریتم است.

$$x[j+1] = x[j] - \sigma^{2-j} y[j] \quad (14)$$

$$y[j+1] = y[j] + \sigma^{2-j} x[j] \quad (15)$$

$$z[j+1] = z[j] - \sigma^{\tan^{-1}(2^{-j})} \quad (16)$$

$$z[0] = \theta \quad (17)$$

$$x[0] = y[0] = \frac{1}{k_h} \quad (18)$$

$$k_h = 0.82816 \quad (19)$$

$$x[f] = y[f] = e^\theta \quad (20)$$

همان‌طور که از روابط (۱۴) تا (۲۰) پیداست، این الگوریتم با استفاده از شیفت‌رجیستر و جمع‌کننده قابل پیاده‌سازی است. برای پیاده‌سازی بخش \tan^{-1} ، با توجه به محدوده z و مقدار \tan^{-1} ، مقادیر آن از قبل محاسبه و ذخیره شدند. در شکل (۷) معماری به کار رفته برای پیاده‌سازی این الگوریتم نشان داده شده است. بیشینه مقدار ورودی برای کارکرد صحیح این الگوریتم برابر است با $\theta = 1.11817$ ؛ به طوری که به ازای مقادیر بزرگ‌تر از آن خروجی درستی بدست نمی‌آید [۳۰].

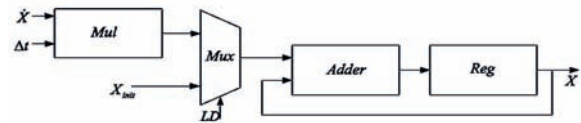
از این‌رو، بنا به درستی روابط (۲۱) و (۲۲) ایده دیگری که از آن

۳-۳- پیاده‌سازی دیجیتال تابع انتگرال گیر

با توجه به شکل (۳) در بخش قبل، ابتدا مقادیر n, V, h, m باید با مقادیر اولیه مقداردهی شوند و بعد از نخستین اجراء خروجی انتگرال بدست آید. برای پیاده‌سازی انتگرال گیر از ساده شده روش اولبر با گام‌های زمانی بسیار کوچک که از زمان آماده شدن پاسخ نورون بدست آمده، استفاده شد. رابطه استفاده شده برای پیاده‌سازی انتگرال گیر در عبارت (۲۳) نشان داده شده است.

$$x(t + \Delta t) = x(t) + \Delta t \cdot \dot{x}(t) \quad (23)$$

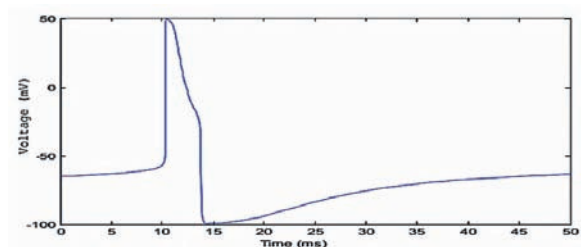
با توجه به رابطه (۲۳) پیاده‌سازی انتگرال گیر با استفاده از مدار شکل (۱۰) قابل انجام است.



شکل ۱۰: بلوک دیاگرام معماری پیشنهادی برای پیاده‌سازی انتگرال گیر

۴- نتایج شبیه‌سازی و پیکربندی محیط آزمایش

برای ارزیابی درستی خروجی نتایج مربوط به شبیه‌سازی پیاده‌سازی دیجیتال با شبیه‌سازی انجام شده با کمک نرم‌افزار MATLAB و در محیط Simulink مقایسه شده است. برای پیاده‌سازی بخش‌های دیجیتال از نرم‌افزار Xilinx ISE بهره گرفته شده است. کلیه شبیه‌سازی‌های انجام شده نیز در نرم‌افزار Modelsim صورت گرفته است. برای شروع پیاده‌سازی و بررسی مرحله به مرحله درستی آن، باید یک مدل پایه وجود داشته باشد، که بتوان مقادیر داده‌های پیاده‌سازی شده را با مقادیر آن سنجید. به این منظور تک نورون مدل H-H در Simulink شبیه‌سازی شد. پس از شبیه‌سازی سطح بالای انجام شده، با اعمال ورودی‌های مختلف به این نورون و بررسی تأثیر تغییرات این ورودی، از درستی عملکرد و شبیه‌سازی اطمینان حاصل شد. شکل (۱۱) نمودار سطح ولتاژ غشاء نورون شبیه‌سازی شده را به ازای جریان ورودی ثابت $I_{ext} = 0.5 A$ نشان می‌دهد. نمودار تغییرات مقادیر درجه‌های n, m و h که سرعت تغییرات این درجه‌ها را نشان می‌دهد، به ترتیب در شکل (۱۲)، شکل (۱۳) و شکل (۱۴) آورده شده است.



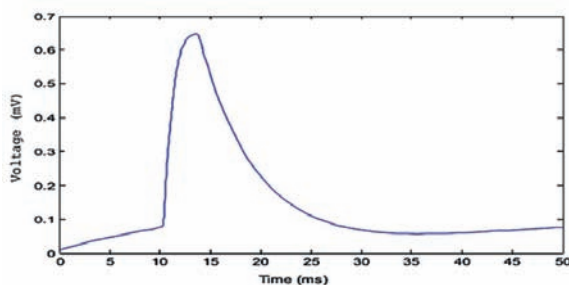
شکل ۱۱: نمودار سطح ولتاژ غشاء نورون به ازای جریان خارجی $I = 0.5 mA$

با بررسی شکل (۱۱)، شکل (۱۲)، شکل (۱۳) و شکل (۱۴) معلوم می‌شود که با اعمال جریان ورودی، ابتدا درجه m با سرعت زیاد فعال شده و باعث افزایش ولتاژ غشاء می‌شود. سرعت باز شدن این درجه بیشتر از تغییرات درجه‌های n و h است. بنابراین قبل از این که درجه‌های n و h بتوانند مانع تغییرات ناشی از درجه m شوند، ولتاژ غشاء نورون به حد آستانه رسیده و نورون آتش می‌کند. سپس با بیشتر شدن مقدار n و کمتر شدن مقدار h سطح ولتاژ

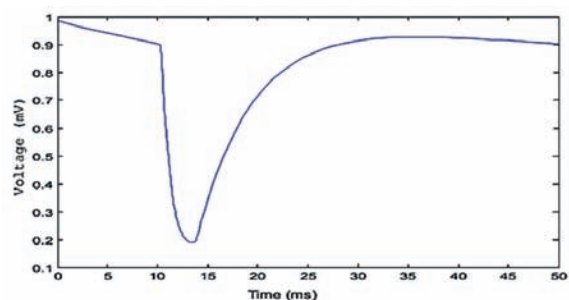
کاهش می‌یابد و این عملیات تازمانی که جریان اعمالی وجود داشته باشد، تکرار می‌شود، نمودار سطح ولتاژ غشاء نورون بدون حضور جریان تحریکی در شکل (۱۵) نشان داده شده است. با توجه به این شکل، در زمان‌های اولیه شبیه‌سازی سطح ولتاژ غشاء نورون به دلیل مقادیر اولیه و برای ایجاد تعادل بین آن‌ها، تغییر کرده، ولی با ثابت شدن مقادیر پارامترها، ولتاژ غشاء نورون نیز ثابت می‌ماند. در شکل (۱۶) نمودار سطح ولتاژ غشاء نورون به ازای جریان تحریکی خارجی ثابت $I = 0.7 mA$ و ولتاژ اولیه $V = -65 mV$ و متغیر درجه اولیه $n = 0.01$ نشان داده شده است. به ازای جریان تحریکی اعمال شده، نورون به صورت تکرار شونده آتش می‌کند. بنابراین درستی داده‌های تک نورون ثابت می‌شود.



شکل ۱۲: نمودار تغییرات متغیر درجه m برحسب زمان به ازای جریان خارجی $I = 0.5 mA$



شکل ۱۳: نمودار تغییرات متغیر درجه n برحسب زمان به ازای جریان خارجی $I = 0.5 mA$



شکل ۱۴: نمودار تغییرات متغیر درجه h برحسب زمان به ازای جریان خارجی $I = 0.5 mA$

در طول مراحل پیاده‌سازی، هر کدام از توابع تشکیل دهنده تک نورون، به‌طور جداگانه و به ازای بازه‌های ورودی مختلف تست شدند و مقادیرشان با مقادیر شبیه‌سازی شده مقایسه شد. در شکل (۱۷) شبیه‌سازی نورون پیشنهادی با نرم‌افزار Modelsim آمده است و درستی عملکرد آن بررسی شده است.

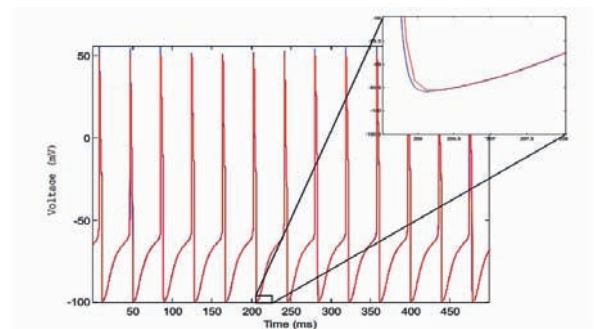
سیستم ممیز شناور و با دقت 64 Bit double انجام شده در حالی که برای دقت مورد نظر 32 bit fix point در نظر گرفته‌ایم که ۱۲ بیت صحیح و ۲۰ بیت اعشار می‌باشد. در جدول (۲) مقایسه زمانی و دقت در نرم‌افزار مطلب و تراشه ARTIX-7 را مشاهده می‌کنید.

جدول ۲: مقایسه زمانی Artix-7 و Matlab

	Artix-7	Matlab
Speed	4 ns	7 μ s
Accuracy	fix point	floating point

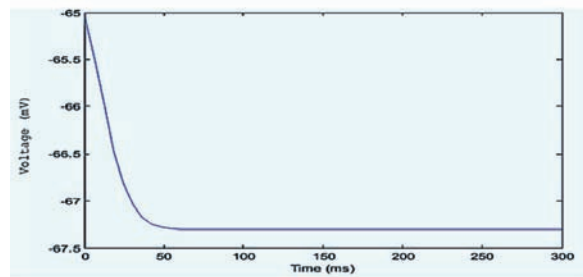
در شکل (۱۸) نمودار سطح ولتاژ غشا تک نورون به ازای جریان ورودی $I = 0.7m A$ و مقدار اولیه $n = 0.01$ نشان داده شده است. نمودار قرمز رنگ نشان دهنده نتایج حاصل از شبیه‌سازی Matlab و نمودار آبی رنگ نشان‌دهنده نتایج حاصل از پیاده‌سازی است. همان‌طور که در این نمودار مشخص است، هیچ تغییری در فرکانس آتش کردن نورون که از جمله خصوصیات مهم در رفتار نورون است، دیده نمی‌شود.

با توجه به شکل (۱۸) تنها اختلاف اندک در دو نمودار است که آن هم از خطای دیجیتال سازی ناشی می‌شود و قابل چشم‌پوشی است. هر چند به علت محدودیت های فضای مربوط به تراشه مورد نظر، امکان موازی سازی کامل در سیستم وجود ندارد ولی در طراحی دیجیتال مورد نظر برای پیاده سازی سیستم نورونی، حداکثر موازی سازی به کار گرفته شد. این موضوع سبب شده است که با توجه به فرکانس کاری سیستم دیجیتال پیاده سازی شده، سرعت عملکرد آن بسیار بیشتر از سرعت اجرای آن بر روی نرم‌افزار باشد. به طوریکه زمان اجرایی سیستم نزدیک به زمان واقعی است، در حالی که زمان شبیه‌سازی سیستم با استفاده از MATLAB بر روی پردازنده Intel Core i5 2.8GHz برابر ۷۳۶/۸۱ ثانیه است. در مشابه‌ترین مقاله، تک نورونی با روش Izhikevich [۳۱] پیاده‌سازی شده است، با توجه به اینکه محاسبات مدل H-H بسیار پیچیده‌تر از سایر مدل‌های نورونی برای دستیابی به دقت بالا می‌باشد اما نسبت به مدل IZH که از محاسبات بسیار ساده‌ای برخوردار است، سرعت محاسبات که ملاک مهمی در عملکرد نورون است، بیشتر است.

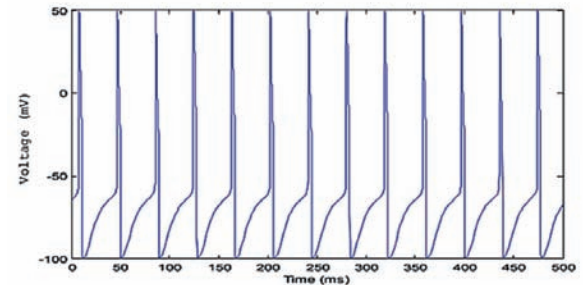


شکل ۱۸: نمودار سطح ولتاژ غشا نورون در حضور جریان تحریکی خارجی (نتایج حاصل از نرم‌افزار Matlab (قرمز)، نتایج حاصل از پیاده‌سازی (آبی))

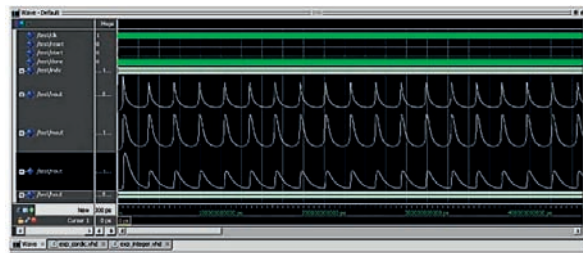
همان‌طور که در جدول (۳) مشاهده می‌شود، استفاده از تراشه Artix-7 به دلیل برخورداری از بلوک‌های DSP، بالا بردن سرعت محاسبات و کاهش سایر منابع سخت‌افزاری، مناسب‌ترین گزینه برای پیاده‌سازی این طرح بوده است که در معماری‌های جدید نیز به آن تاکید شده است [۳۲، ۳۳].



شکل ۱۵: نمودار سطح ولتاژ غشا نورون بدون حضور جریان تحریکی



شکل ۱۶: نمودار سطح ولتاژ غشا نورون در حضور جریان تحریکی ثابت



شکل ۱۷: بخشی از شبیه‌سازی اسپایک‌های نورون پیشنهادی در نرم‌افزار Modelsim

۵- گزارش پیاده‌سازی و ارزیابی عملکرد سیستم بر حسب زمان

در این بخش درستی سیستم پیاده‌سازی شده با استفاده از داده‌های معتبر بدست آمده از شبیه‌سازی بخش قبل، مورد بررسی قرار گرفته و به تحلیل عملکرد سیستم، تحلیل زمانی و سخت‌افزار مصرفی پرداخته شده است. در جدول (۱) نتایج سنتز مدل پیشنهادی و منبع سخت‌افزاری قابل دسترس در تراشه مورد نظر آمده است.

جدول ۱: گزارش سنتز مدل پیشنهادی

	HH(ARTIX-7)	
	used	available
FF	1700	160000
RAM	700byte	13Mbyte
LUT	5400	215000
DSP BLUK	4	740
MAX SPEED	250MHZ	

در محاسبات Matlab برای به‌روزرسانی گام‌ها از مدت زمانی که برابر است با ۷ میکروثانیه استفاده می‌شود اما در سخت‌افزار موجود بیشترین کلاک برابر با ۴ نانوثانیه اندازه‌گیری گردید بدون اینکه مدار از مقدار زمانی که دارد، تخطی کند. در Matlab تمامی محاسبات با

جدول ۳: مقایسه سخت‌افزاری و سرعت

	IZH(VIRTIX-II)		HH(ARTIX-7)	
	used	available	used	available
بلوک DSP	----	----	4	740
سرعت ماکزیمم	241MHZ		250MHZ	

۶- نتیجه‌گیری

در این مقاله به پیاده‌سازی یک سیستم نوروئی زیستی مدل H-H مبتنی بر FPGA پرداخته شد. در مقاله حاضر در پیاده‌سازی نوروئی از روش ساده‌سازی محاسباتی بهره برده و با ایده‌های دیگر در طراحی، دقت محاسبات را افزایش دادیم. همچنین برای غلبه بر محدودیت‌های عملکرد نرم‌افزاری، طراحی سخت‌افزار با سرعت و با عملکرد بالا انجام شد. بدین منظور نیز ایده‌های مختلفی در طول مراحل پیاده‌سازی در نظر گرفته شد که با به کار گرفتن آنها، تک نوروئی پیاده‌سازی شده دارای سرعت بسیار بالا و زمان اجرایی نزدیک به زمان واقعی باشد. از جمله ویژگی‌های دیگر این طرح، مقیاس‌پذیر و قابل توسعه بودن آن برای تعداد بیشتری از نوروئی‌ها است. از مدل بهبودیافته در این مقاله می‌توان برای ساخت یک بستر مناسب که با الهام از ساختار مغز بتواند قابلیت‌های مغز نظیر حجم کوچک، سرعت پردازش زیاد، مصرف توان کم و قابلیت تشخیص و استنتاج چشمگیری را در خود داشته باشد، استفاده نمود.

مراجع

- FPGA implementation of a neural network for character recognition," *Advances in Neural Networks Symposium*, pp. 1357–1365, China, 2006.
- [14] S. Sahin, Y. Becerikli, and S. Yazici, "Neural network implementation in hardware using FPGAs," in *Neural Information Processing*, pp. 1105–1112, China, 2016.
- [15] Q. Zou, Y. Bornat, J. Tomas, S. Renaud, and A. Destexhe, "Real-time simulations of networks of Hodgkin-Huxley neurons using analog circuits," presented at the *Neurocomput*, vol. 69, pp. 1137–1140, 2006.
- [16] E. L. Graas, E. A. Brown, and R. H. Lee, "An FPGA-based approach to high-speed simulation of conductance-based neuron models," *Neuroinformatics*, vol. 2, pp. 417–435, 2004.
- [17] P. Pourhaj and D. H.-Y. Teng, "FPGA based pipelined architecture for action potential simulation in biological neural systems," *Conference on Electrical and Computer Engineering*, pp. 1–4, Canada, 2010.
- [18] Y. Zhang, J. Nunez-Yanez, J. McGeehan, E. Regan, and S. Kelly, "A biophysically accurate floating point somatic neuro processor," *Conference on Field Programmable Logic and Applications*, pp. 26–31, Prague, 2009.
- [19] S. Saighi, J. Tomas, Y. Bornat, B. Belhadji, O. Malot, and S. Renaud, "Real-time multi-board architecture for analog spiking neural networks," *Proceedings of 2010 IEEE International Symposium on Circuits*, pp. 1939–1942, France, 2010.
- [20] S. Saighi, T. Levi, B. Belhadji, O. Malot, and J. Tomas, "Hardware system for biologically realistic, plastic, and real-time spiking neural network simulations," *IEEE International Joint Conference on Neural Networks*, pp. 1–7, Spain, 2010.
- [21] F. Grassia, T. Lévi, J. Tomas, S. Renaud, and S. Saighi, "A neuromimetic spiking neural network for simulating cortical circuits," *IEEE Information Sciences*, pp. 1–6, USA, 2011.
- [22] M. Mokhtar, D. Halliday, and A. Tyrrell, "Hippocampus-inspired spiking neural network on FPGA," *Evolvable Systems: From Biology to Hardware*, pp. 362–371, Prague, 2008.
- [23] K. L. Rice, M. A. Bhuiyan, T. M. Taha, C. N. Vutsinas, and M. C. Smith, "FPGA Implementation of Izhikevich Spiking Neural Networks for Character Recognition," *IEEE in International Conference on Reconfigurable Computing and FPGAs*, pp. 451–456, Mexico, 2009.
- [24] A. Cassidy and A. G. Andreou, "Dynamical digital silicon neurons," *IEEE Biomedical Circuits and Systems Conference*, pp. 289–292, USA, 2008.
- [25] E. M. Izhikevich, "Which model to use for cortical spiking neurons?," *IEEE Transactions on Neural Networks*, vol. 15, pp. 1063–1070, 2004.
- [26] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, pp. 500–544, 1952.
- [27] C. Börgers, S. Epstein, and N. J. Kopell, "Background gamma rhythmicity and attention in cortical local circuits: a computational study," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, pp. 7002–7007, 2005.
- [28] G. B. Ermentrout and N. Kopell, "Fine structure of neural spiking and synchronization in the presence of conduction delays," *Proceedings of the National Academy of Science*, vol. 95, pp. 1259–1264, 1998.
- [29] M. D. Ercegovac and T. Lang, *Digital arithmetic*, Morgan Kaufmann publishes, 2003.
- [30] D. R. Llamocca-Obregón and C. P. Agurto-Ríos, "A fixed-point implementation of the expanded hyperbolicCORDIC algorithm," *Latin American applied research*, vol. 37, pp. 83–91, 2007.
- [31] H. Soleimani, A. Ahmadi, and M. Bavandpour, "Biologically Inspired Spiking Neurons," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, pp. 2991–3004, 2012.
- [32] Q. Y. Sun, Q. X. Wu, X. Wang, L. Hou, "A spiking neural network for extraction of features in color opponent visual pathway and FPGA implementation" *Int. J. Neurocomputing*, vol. 228, pp. 119–132, 2017.
- [33] L. Wan, Y. Luo, "Efficient neuron architecture for FPGA based spiking neural network" *IEEE Signals and Systems*, pp. 1–6, UK, 2016.
- [1] E. Kandel, J. Schwartz, and T. Jessell, *Principles of Neural Science*, 4nd edition., McGraw-Hill Medical, 2000.
- [2] J. Misra and I. Saha, "Artificial neural networks in hardware: A survey of two decades of progress," *Neurocomputing. Mag.*, vol. 74, no. 1, pp. 239–255, 2010.
- [3] M. A. Cavu, C. Karakuzu, S. Sahin, and M. Yakut, "Neural network training based on FPGA with floating point number format and it's performance," *Neural computing & applications. Mag.*, vol. 6, no. 1, pp. 195–202, 2011.
- [4] Y. Heo and H. Song, "Circuit modeling and implementation of a biological neuron using a negative resistor for neuron chip," *BioChip J*, vol. 92, no. 2, pp. 17–24, 2012.
- [5] A. Muthuramalingam, S. Himavathi, and E. Srinivasan, "Neural network implementation using FPGA: issues and application," *International journal of information technology*, vol. 4, no. 2, pp. 86–92, 2008.
- [6] F. J. Pelayo, E. Ros, X. Arreguit, and A. Prieto, "VLSI Implementation of a Neural Model Using Spikes," *Analog Integrated Circuits and Signal Processing*, vol. 13, no. 1, pp. 111–121, 1997.
- [7] L. Gatet, H. Tap-Béteille, and F. Bony, "Comparison between analog and digital neural network implementations for range-finding applications," *Neural Networks, IEEE Transactions*, vol. 20, no. 3, pp. 460–470, 2009.
- [8] L. Wanhammar, *DSP Integrated Circuits*, 2nd edition., Academic Press, 1999.
- [9] G. Li, V. Talebi, A. Yoonessi, and C. L. Baker Jr, "A FPGA real-time model of single and multiple visual cortex neurons," *J. Neurosci. Methods*, vol. 193, no. 1, pp. 62–66, 2010.
- [10] E. M. Izhikevich, *Dynamical systems in neuroscience: the geometry of excitability and bursting*, 1st ed. MIT press, 2006.
- [11] J. A. Bailey, R. Wilcock, P. R. Wilson "Behavioral simulation and synthesis of biological neuron systems using synthesizable VHDL," *Neurocomputing*, vol. 74, pp. 2392–2406, 2011.
- [12] M. Bahoura and C. W. Park, "FPGA-implementation of dynamic time delay neural network for power amplifier behavioral modeling," *Analog Integrated Circuits and Signal Processing*, vol. 73, no. 3, pp. 819–828, 2012.
- [13] F. Khan, M. Uppal, W. C. Song, M. J. Kang, and A. Mirza,
